SESSION INITIATION PROTOCOL
FOR WIRELESS CHANNELS

A Thesis

by

VIJAY SUNDAR RAJARAM

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2006

Major Subject: Electrical Engineering

SESSION INITIATION PROTOCOL FOR WIRELESS CHANNELS

A Thesis

by

VIJAY SUNDAR RAJARAM

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Scott L. Miller
Committee Members,	Don R. Halverson
	Andrew K. Chan
	Paul Nelson
Head of Department,	Costas Georghiades

December 2006

Major Subject: Electrical Engineering

ABSTRACT

Session Initiation Protocol for Wireless Channels. (December 2006)

Vijay Sundar Rajaram, B.E., Bharathiyar University, PSG College of Technology, India

Chair of Advisory Committee: Dr. Scott L. Miller

The Session Initiation Protocol (SIP) was designed for wire line networks. It was developed to initiate, modify and terminate sessions between two hosts on a network. When the Internet expanded to include wireless hosts, SIP did not scale well for these wireless hosts because of the nature of the wireless channel. Also, there were issues with mobility and real time communication. This thesis proposes improvements to some of the extensions to SIP, for better performance over wireless channels. We investigate the call setup time for various transport mechanisms viz. TCP and UDP, and study the performance of a dynamic Session Timers compared to the current standard of a periodic refresh mechanism, where the frequency of `UPDATE`s vary with the condition of the wireless channel. We also propose a handoff algorithm that reduces the handover time with decreased packet losses.

To my loving parents and my brother

ACKNOWLEDGMENTS

I would like to thank my committee chair, Dr. Scott L. Miller for his insight, never ending support, invaluable guidance and constant discussions that helped me to achieve this goal. I also would like to thank my committee members Dr. Don R. Halverson, Dr. Andrew K. Chan and Dr. Paul Nelson for their support and time. I would like to thank the Head of the Electrical Engineering Department, Dr. Costas Georghiades, for providing me an opportunity to present this research.

I would like to thank the Electrical Engineering Department and Texas A&M University for providing the necessary infrastructure to carry out this research. I would like to extend my gratitude to the Internet2 Technology Evaluation Center, Texas A&M University and to the Director of Telecommunications, Dr. Walt Magnussen, for their support.

Special thanks go to professors in the Wireless Communications group for their excellent teaching. I have learned much more from both course work and the process of doing research.

I also extend my thanks to my parents and relatives for their continued encouragement and moral support. Finally, I thank my friends for their timeless help and support throughout the period of my research.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Internet telephony – voice transmission and call signalling over IP networks – has in recent years become an area of intense development. As IP networks have become ubiquitous, it has become increasingly clear that having separate networks for voice and data communications is unnecessary and redundant. Circuit-switched telephone networks, however, are complicated systems. As Internet telephony systems have been developed and deployed, experience has revealed many features of circuit-switched networks which the initial versions of IP telephony systems did not completely replicate. Thus, much of the recent development work on these systems, following the initial development of the basic protocols, has been to create systems which allow IP telephony systems to replicate features, architectures, and attributes of circuit-switched networks. For example, recent work has enabled IP telephony systems to offer call transfer services; to allow end systems to communicate with automated systems using Touch-Tone (DTMF) tones; and to ensure quality of service and reliability of voice transport over potentially unreliable data networks.

This is all interesting and important work, and essential to ensure that Internet telephony can provide a satisfactory replacement for circuit-switched telephone networks. However, Internet telephony can provide services far beyond those of the circuit-switched network, exactly because it is on the Internet. The Internet environment is different from telephone networks in two fundamental ways. First of all, it is not limited to a single form of communication. Internet end systems can simultaneously be reading e-mail, browsing web pages, and sending instant messages, as well as

---

The journal model is *IEEE Transactions on Automatic Control.*

communicating by voice. Secondly, the Internet is decentralized. Every end system can (in principle) communicate with every other end system; intermediate devices are only necessary if they provide some service to the end systems. The increase in the available bandwidth, has resulted in increased popularity of voice over IP communications in the wire line segments. The advent of 3G wireless communications has greatly increased the bandwidth available for data communications over wireless channels. Hence VoIP over wireless channels is gaining increasing interest.

In cellular communications, voice and data traffics are usually treated separately. With the deployment of VoIP, it is possible to send voice over data channels (IP based packet switched networks). However, unlike the wire line links, the wireless link is error prone. There are errors due to fading and shadowing, resulting in a high Frame error rate (FER), as much as 10 %. In order to cope with the high FER, the Data link layer in wireless networks include a Radio Link Protocol (RLP)[1] sub-layer along with the MAC and Logical link sub-layers. RLP was proposed to provide extra reliability to wireless networks.

An important aspect of telephony services is signaling. For VoIP, two signaling schemes are popular. The first one is H.323[2], specified by ITU-T for the implementation of multimedia services. It is not a single standard but is an umbrella of standards for video conferencing. The second standard is Session Initiation Protocol (SIP)[3] developed by Internet Engineering Task Force (IETF). It is an application layer signalling protocol that can establish, modify and terminate multimedia sessions. SIP is developed exclusively for the Internet unlike H.323 and is similar to `HTTP` protocol [4] and `SMTP` [5] in its structure.

This research focusses on the IETF standard for media signaling, the Session Initiation Protocol (SIP) [3]. SIP was designed for wire-line networks and was designed to scale well with the Internet. However, with the growth of wireless data

services, SIP signaling was found to be inefficient for wireless services. This research studies the effect of wireless channel on SIP Session startup time and proposes certain improvements for Session Timers and handoff latency.

A. Related Work

There has been a lot of interest in adapting SIP to wireless networks ever since the draft standard was accepted in 1999. The interests were in the area of mobility management for wireless 3G networks [6] [7] [8] [9] [10] [11] and application layer handovers [12] [13] [14] [15] [16] [17]. There have been extensions to SIP [18] [19]to provide additional functionality but these extensions were not targeted towards wireless networks. We see that there has been considerable interest in using SIP for handovers and mobility management. However, there has not been efforts toward adapting SIP and its extensions to wireless environment. In this research, we do not modify the SIP specification (RFC 3261) but we propose some changes to a couple of SIP extensions for wireless channels.

B. Thesis Objective and Scope

In this research, we focus on the adaptation of the Session Initiation Protocol for wireless channels. The research is divided into three sections. First, we investigate the session setup times under different protocols and determine their performances under different wireless channel error conditions. Second, we adapt the SIP Session Timer for wireless channels by making the frequency of `UPDATE`s dependent on the channel conditions. Finally, we investigate the performance of a hybrid handover technique that seeks to achieve faster handoff times compared to existing techniques.

C. Thesis Outline

The chapters are organized as follows. Chapter II provides an introduction to the Session Initiation Protocol and explains its working. Chapter III discusses the session startup times under different protocols. Chapter IV explains the SIP Session Timer and discusses the performance of the dynamic adaptation scheme through an analytical model.Chapter V describes the implementation of the hybrid handover algorithm, the simulation setup and the results obtained. Chapter VI concludes the research work.

CHAPTER II

SESSION INITIATION PROTOCOL

Session Initiation Protocol (SIP)[3] is an application-layer signaling protocol that can establish,modify, and terminate multimedia sessions (conferences) such as Internet telephony calls. SIP can also invite participants to already existing sessions, such as multi-cast conferences. Media can be added to (and removed from) an existing session. SIP transparently supports name mapping and redirection services,which supports personal mobility - users can maintain a single externally visible identifier regardless of their network location. SIP supports five facets of establishing and terminating multimedia communications:

- User location: Determination of the end system to be used for communication;
- User availability: determination of the willingness of the called party to engage in communications;
- User capabilities: Determination of the media and media parameters to be used;
- Session setup: "Ringing", establishment of session parameters at both called and calling party;
- Session management: Including transfer and termination of sessions, modifying session parameters, and invoking services.

A. Description

A detailed description of SIP can be found in [3]. We briefly overview it here, providing a broad understanding of the protocol and the functioning of its components.

SIP is a client-server protocol, similar in both syntax and semantics to the Hyper Text Transfer Protocol (HTTP). However, it defines its own methods and headers for providing the functions required in IP telephony signaling. Requests are generated by one entity (the client), and sent to a receiving entity (the server) that processes them, and then sends responses. A request and the responses which follow it are called a *transaction*. As a call participant may either generate or receive requests, SIP-enabled end systems include a protocol client and server (a user agent client and user agent server respectively). The user agent server responds to the requests based on human interaction or some other kind of input. Furthermore, SIP requests can traverse many proxy servers, each of which receives a request and forwards it towards a next hop server, which may be another proxy server or the final user agent server. As a result, proxy servers are primarily responsible for call routing. A server can make use of any means at its disposal to determine where to route a call, including database queries or local program execution.

A server may also act as a redirect server, informing the client of the address of the next hop server, so that the client can contact it directly. There is no protocol distinction between a proxy server, redirect server, and user agent server; a client or proxy server has no way of knowing which it is communicating with. The distinction lies only in function. A proxy or redirect server cannot accept or reject a request, whereas a user agent server can. This is similar to the HTTP model of clients, origin and proxy servers. A single host may well act as client and server for the same request.

1. Comparison with HTTP

As in `HTTP`, the client requests invoke methods on the server. Requests and responses are textual, and contain header fields which convey call properties and service information. SIP reuses many header fields used in `HTTP`, such as the entity headers (e.g.,

Content-Type) and authentication headers.

Like HTTP, SIP requests and responses carry bodies, which can be any defined `MIME` [21] type. The body conveys information about the session between the parties. Normally, the Session Description Protocol (SDP) [22] is used. SDP describes multimedia sessions, including codec types, port numbers, addresses, and session start and stop times.

SIP runs on top of UDP, TCP or SCTP, providing its own reliability mechanisms when used with UDP. For addressing, SIP makes use of uniform resource identifiers (URIs), which are generalizations of Uniform Resource Locators (URLs), in common usage in the web. SIP defines its own URI, but its header fields can carry other URIs, such as http, `mailto` [23], or `tel`[24] URLs.

Like HTTP, a proxy or redirect server can destroy all transaction state after the transaction is complete. SIP transactions can complete even if a server crashes and reboots in the middle, losing all transaction state. SIP messages contain sufficient information to allow a rebooted server to treat the message correctly. This also means a server can safely clean up old state which has collected due to unusual failures or cases where the caller lets the phone ring for a long time. In addition, SIP allows subsequent transactions (such as a call termination, or feature invocation) to occur directly between the caller and callee without traversing through the same proxy or redirect servers. However, a proxy can insist on being in the signaling path for subsequent transactions for the same call through means of the `Route` and `Record-Route` header fields.

2. SIP Methods

SIP defines several methods. `INVITE` invites a user to a call. `BYE` terminates a connection between two users in a call. `OPTIONS` solicits information about capabilities,

but does not set up a connection. `ACK` is used for reliable message exchanges for invitations. `CANCEL` terminates a search for a user. `REGISTER` conveys information about a users location to a SIP server. Many SIP extensions have been written; some of these define other methods as well.

A call is set up by issuing an `INVITE` request. This request contains header fields used to convey information about the call. Examples of some of the header fields are `To`, which lists the callee, `From` which lists the caller, `Subject`, which identifies the subject of the call, `Call- ID` which contains a unique call identifier, `Contact` which lists addresses where a user can be contacted, and `Require`, which allows for feature negotiation.

At any time after the call is set up, either party may send a new `INVITE` request to the other to change parameters of the call. This includes changing media codecs, adding parties to the call, or changing addresses (for mobility support). Once the call is over, either side may hang up by sending a `BYE` request to the other.

B. How SIP Works

Fig.1 illustrates the working of SIP in the presence of a proxy server.

The steps are as follows:

1. The caller sends an `INVITE` message to the callee with its media information embedded in SDP.

2. The message reaches the proxy server, and the proxy server determines how to route the packets to the callee. The proxy server also sends a provisional message `100 Trying` to prevent the caller from sending additional `INVITE`s.

3. The proxy server routes the packet to the callee.

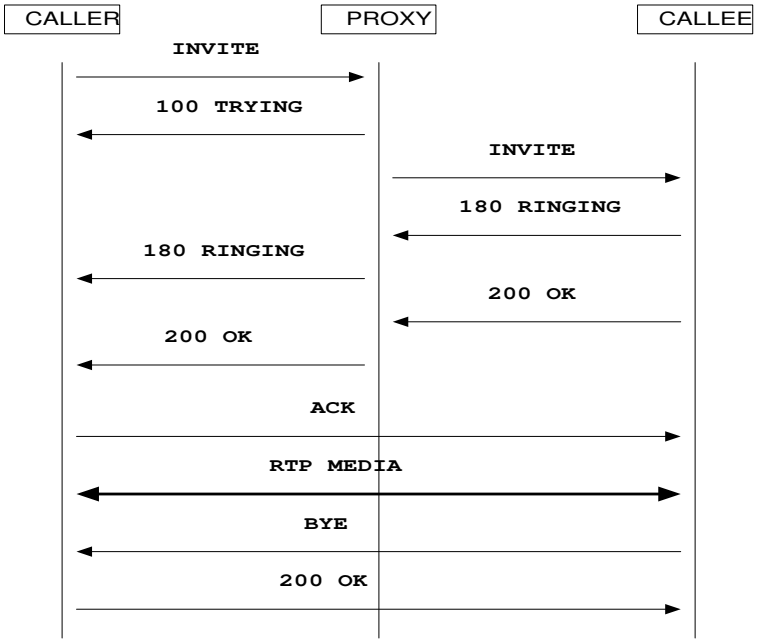

Fig. 1. SIP Call Flow

4. The callee responds with a `180 Ringing` message which is relayed to the caller. The caller's service agent generates the ring back tone.

5. The callee sends a `200 OK` final response to the caller. This response contains the media description of the callee.

6. The caller sends an `ACK` message offering a set of media parameters that will be used in the call.

7. The RTP media stream that follows, is a direct connection between the two endpoints and does not necessarily involve the proxy servers.

8. When either one of the endpoints disconnects the call, a `BYE` request is sent to the other endpoint in the call and it responds with an `200 OK` message. This terminates the call.

In the next chapter we shall discuss the session startup time under TCP and UDP under varying channel error conditions.

CHAPTER III

SESSION STARTUP TIME

"*We define session startup time as the period between the initiation of the INVITE request and the instant of reception of the* `ACK` *at the destination user agent*".

Since SIP supports both TCP and UDP, we seek to determine the session startup time (i.e., the time required for the signalling, before the media communications can begin) under both protocols. Radio Link Protocol [1] is a data-link layer protocol, which provides services for wireless channels. In this chapter, we will investigate the session startup time over TCP and UDP with and without the aid of RLP.

A. SIP Session Establishment

Let us consider the SIP session establishment transaction. The establishment of a session using SIP consists of an `INVITE` transaction and an `ACK` transaction. The caller starts a transaction by sending a SIP INVITE request to callee. The `INVITE` request consists of the details of the type of the session and the capabilities of the caller (via SDP). When the callee accepts the call, a `200 OK` response is sent to the caller to show the agreement on the type of media. The last step in beginning the session is the transmission of the `ACK` request from the caller to the callee. This transaction does not have any responses to its request. Thus the media session is established.

In order to ensure the reliable delivery of the signalling messages, SIP employs a retransmission mechanism. The client side of the transaction involves sending the INVITE request and the ACK request. The server side sends the 200 OK request and receives an ACK as the response.

- When the SIP messages are carried over UDP, the client retransmits the INVITE

message after T_{rc} seconds and doubles after each retransmission

- When the SIP messages are carried over TCP, TCP handles packet losses by setting a timer when it sends the data, and if the packet is not acknowledged when the timer expires, it retransmits the data

But for both TCP and UDP, the retransmission of requests cease after six retransmissions i.e. $2^6 T_{rc}$ seconds. The retransmission mechanism is the same for both the server and the client.

B. Radio Link Protocol

Radio Link Protocol (RLP) [25] is an automatic repeat request (ARQ) protocol used over a wireless (typically cellular) air interface. Most wireless air interfaces are tuned to provide 1% packet loss, which is a tolerable loss rate for modern vocoders. An RLP detects packet losses and performs retransmissions to bring packet loss down to .01%, which is suitable for TCP/IP applications. RLP also implements stream fragmentation and reassembly, and sometimes, in-order delivery. Newer forms of RLP also provide framing and compression, while older forms of RLP rely upon a higher-layer PPP [26] protocol to provide these functions. An RLP transport never knows how big a packet the air interface will provide. Instead, the air interface scheduler determines the packet size, and calls upon RLP to form a packet on-demand for transmission.

An RLP protocol can be ACK-based or NAK-based. Because the reverse link is very expensive on most cellular networks, most RLP's are NAK-based, meaning that the sender assumes that the transmission got through, and the receiver only NAKs when an out-of-order segment is received.

When RLP finds a frame in error (or missing), it sends back a NAK requesting

for the retransmission of only the lost frame and sets a timer. When the NAK reaches the transmitter, it triggers the retransmission of the lost frame. In the event that the receiver timer expires, it retransmits another NAK. This process is continued till the number of retransmission exceeds a limit (n). RLP will now abort the attempt and pass the remaining frames to the upper layers. Further recovery is the responsibility of the upper layers.

During the data transfer phase, RLP maintains the sending sequence number count V(S) and two sequence numbers for receiving, V(R) and V(N). V(S) is incremented by modulo 256 whenever a new frame of non-zero bytes is sent out (it is the sequence number of the next frame expected to be received) and V(N) is the oldest sequence number of the missing frames. If the sequence number of the newly received frame is denoted by SEQ, RLP transmission procedure can be described as follows –

- If SEQ $<$ V(N), or if the frame is already stored in the re-sequencing buffer, discard
- If SEQ = V(N), update V(N) to the next oldest missing frame sequence number. Pass the received frame up to V(N) -1 to the upper layer.
- If V(N) $<$ SEQ $<$ V(R), store frame SEQ in the re-sequencing buffer if it is missing.
- If SEQ = V(N) = V(R), pass all frames received up to V(R) to the upper layer.
- If SEQ = V(R) (V(N) or SEQ $>$ V(R)), increment V(R) and store frame SEQ into re-sequencing buffer.
- Update the NAK list

- For all cases, send NAK's of missing frames if their retransmission timers are not yet set or expired.

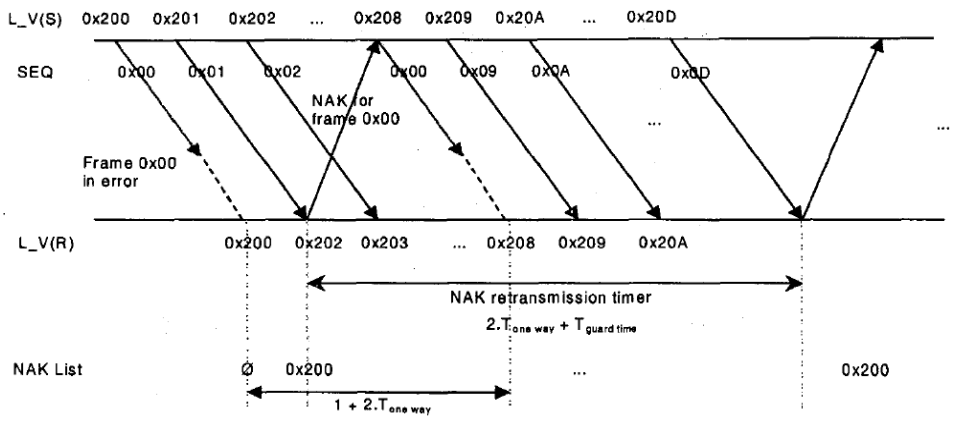

Fig. 2. RLP Operation

The RLP operation is shown in Fig.2. The RLP aborts on a frame after exhausting the number of allowed retransmission attempts and forwards the frames which were waiting for the delayed frame because of the ordered deliver requirement to the application layer. Note that for the purpose of the discussion, NAK retransmission timer and NAK abort timer are the same. The number of NAK's per round and number of rounds of NAK determine the delay characteristics and retransmission scheme and the retransmission overhead. For the purpose of this work on Session setup time, RLP (1,2,3) is employed. RLP (1,2,3) describes the retransmission sequence followed by RLP. If the original transmission of the frame fails (1), RLP attempts to retransmit the frame twice (2). If the second round of retransmission fails, then RLP requests three more copies of NAK from the receiver before aborting the recovery of the frame (3).

Let us now calculate the session startup time for UDP and TCP. We shall subsequently consider the effect of RLP on TCP and UDP with respect to session startup

times.

C. Calculation of Session Startup Time over UDP without RLP

Let us first consider the simple case where the SIP messages are sent over UDP and without the RLP interface for the wireless link. Fig. 3 describes the SIP messages involved in setting up a media connection.

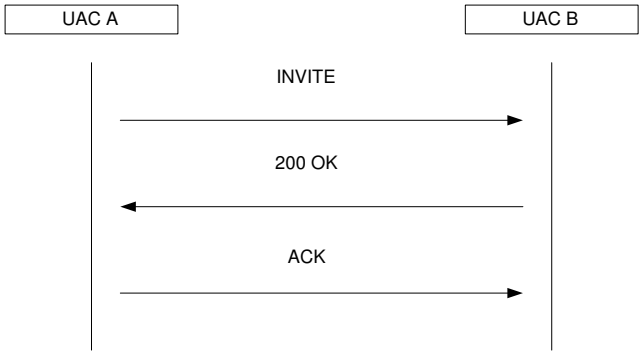

Fig. 3. SIP Messages over UDP

According to our earlier discussion on retransmission mechanism, we have the retransmit timer T_{rc} double after every retransmission, with T_r being the retransmission interval.

$$T_{rc} = 2^{i-1}.T_r \tag{3.1}$$

T_r is an important parameter in the session setup delay.

- If T_r is small, retransmission will begin even when the response is coming back. The decision on the expiry will be made earlier than necessary.
- If T_r is large, the session setup time increases for the wait would be longer than necessary for expiration of the timer.

Notation:

Let l = Number of air link frames in the UDP datagram

τ = Inter-frame time

d= End to end propagation delay

Suppose,

INVITE message has $l1$ frames

200 OK has $l2$ frames

ACK has $l3$ frames

Client Side Retransmission timer

$$T_{rc} = d + (l1 - 1)\tau + (l2 - 1)\tau + d \tag{3.2}$$

Server side Retransmission timer

$$T_{rs} = d + (l2 - 1)\tau + (l3 - 1)\tau + d \tag{3.3}$$

Let p = probability that a frame is lost in the air link.If a UDP packet consists of k frames, assuming the packet losses are independent

packet loss rate = $1 - (1 - p)^k$

If q = probability of retransmission, then for a client side transaction (e.g. `INVITE`), q is the probability that the transaction fails i.e. the first packet (`INVITE`) fails or `INVITE` is sent successfully and the `200 OK` fails

On the client side,

$$q_c = 1 - (1 - p)^{l1+l2} \tag{3.4}$$

On the server side,

$$q_s = 1 - (1 - p)^{l2+l3} \tag{3.5}$$

If N is the maximum number of transmissions

Average delay for a successful transaction is the average delay for successfully transmitting the UDP datagram containing a SIP message and successfully receiving the responses.

Let the normalized delay for the i^{th} UDP datagram $= T_N(i)_{UDP}$

$$T_N(i)_{UDP} = \frac{1}{1-q^N}\{(1-q)(d+(l-1)\tau) + (1-q)(T_r + d + (l-1)\tau) + \\ ... + (1-q)q^{N-1}((2^{N-1}-1)T_r + d + (l-1)\tau)\} \quad (3.6a)$$

Simplifying,

$$T_N(i)_{UDP} = d + (l-1)\tau - T_r + \frac{(1-q)(1-(2q)^N)}{(1-q^N)(1-2q)}T_r \quad (3.6b)$$

$$T_{UDP} = \sum_{i=1}^{N} T_{iUDP} \quad (3.7)$$

D. Calculation of Session Startup Time over TCP without RLP

Consider the messages that are transmitted across the two endpoints in a TCP connection in Fig.4

The propagation time is the same as in UDP, though in the event of loss, TCP's overcautious approach reduces its window size to half of what it was before the loss. But the rate of tranmission is not the primary concern here. So the expressions for call setup time derived for UDP holds good for TCP too with the exception of N, which is the maximum number of transmissions. N is larger for TCP owing to the larger number of messages sent and retransmitted in the event of loss.

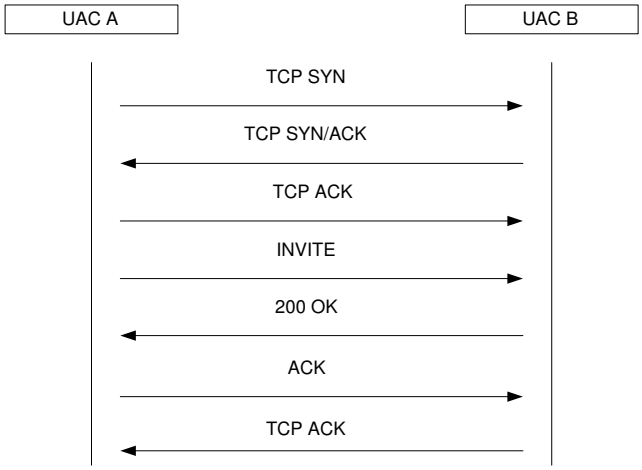

Fig. 4. SIP Messages over TCP

$$T_N(i)_{TCP} = \frac{1}{1-q^N}\{(1-q)(d+(l-1)\tau) + (1-q)(T_r + d + (l-1)\tau) + \\ ... + (1-q)q^{N-1}((2^{N-1}-1)T_r + d + (l-1)\tau)\} \quad (3.8a)$$

Simplifying,

$$T_N(i)_{TCP} = d + (l-1)\tau - T_r + \frac{(1-q)(1-(2q)^N)}{(1-q^N)(1-2q)}T_r \quad (3.8b)$$

$$T_{TCP} = \sum N_{i=1} T_{iTCP} \quad (3.9)$$

E. Calculation of Session Startup Time over TCP and UDP with RLP

The use of RLP provides a faster startup time for both TCP and UDP, because the retransmission of a smaller frame is faster than the retransmission of a larger packet. The frame delay of TCP with RLP was derived by Gang Bao in his work on RLP over CDMA wireless channels [27]. The expressions are the same for both TCP and UDP. But for TCP, the number of frames transmitted will be higher than UDP since

TCP is a reliable transport protocol. From his results in [27], we have the average frame delay calculated as

$$T = d(1-p) + \sum_{i=1}^{n} \sum_{j=1}^{i} (2id + 2(j-1)C_{i,j}\tau) \tag{3.10}$$

where, d is the propagation delay in sending the frames, p is the FER, and $C_{i,j}$ refers to the first frame that was received correctly at the destination, which is the ith retransmitted frame at the jth retransmission trail, and $C_{i,j}$ was computed to be

$$C_{i,j} = p(1-p)^2((2-p)p)^{\frac{(i^2-i)}{2}+j-1}$$

F. Numerical Results

This section presents the calculations of the average session setup delay for SIP over both TCP and UDP. From the model in the previous sections, we see a direct relationship between FER and session setup delay. The number of transactions and the size of the message affects the average session setup delay. In order to evaluate the approximate size of each SIP message, using Ethereal [28],we capture some of the SIP packets and determine the average number of frames at 9.6 kbps to be 37 per message. The inter frame delay, τ is taken as 20 ms and the end to end propagation delay, d is 100 ms respectively.

From Fig.5, we see that the use of RLP greatly reduces the session setup time in the case of TCP. While we cannot tolerate the high session setup time in the case of TCP without RLP, the use of RLP has brought the session setup time to within acceptable limits. The reliability is provided by the RLP retransmission timer. When RLP is not able to correctly receive the frames within the stipulated period, it is up to the higher layer(TCP) to provide reliability. In the case of UDP, the protocol does not retransmit the packets. But SIP continues to retransmit the message six times

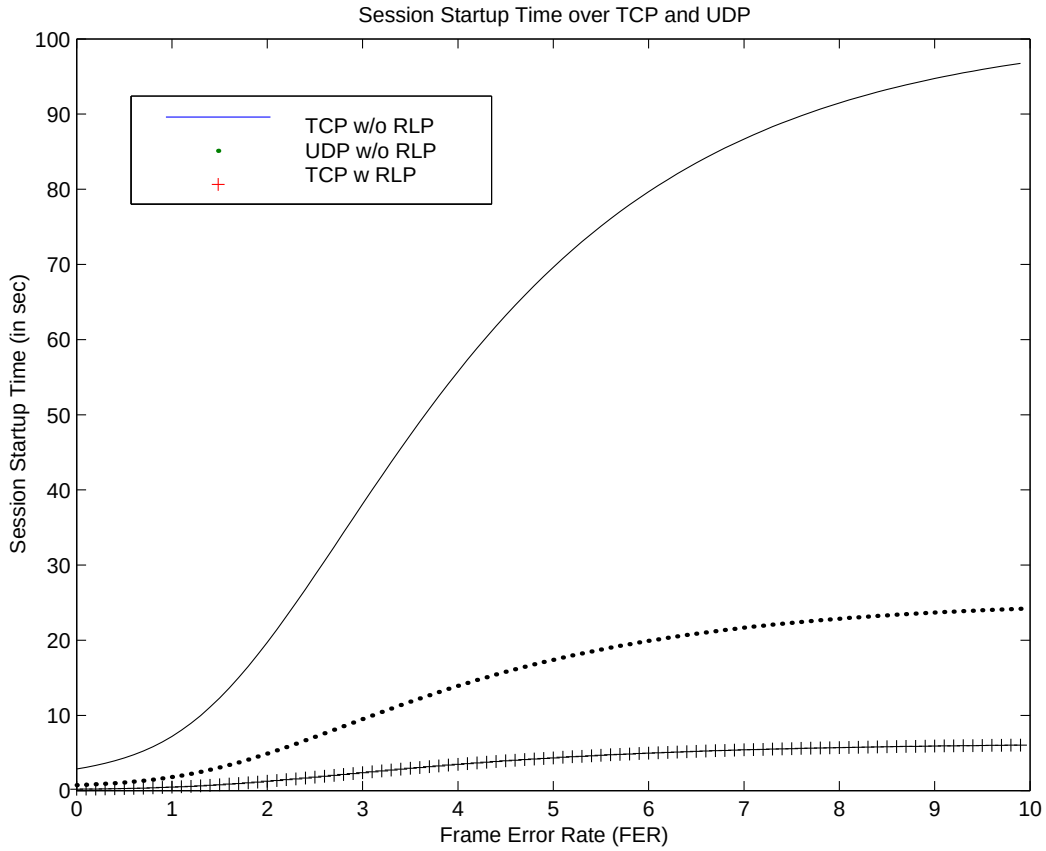

Fig. 5. Comparison of SIP Session Startup Time over TCP and UDP

till the session is aborted.

We observe that with RLP, TCP does provide an acceptable session startup time for a reliable establishment of an session.

In the next chapter, we will discuss the performance of a dynamic session timer for a wireless channel.

CHAPTER IV

SIP SESSION TIMER

The basic SIP specification does not define a mechanism for proxy servers to track the states of the sessions after the sessions are established [3]. In other words, a basic SIP Proxy server is not able to determine whether an established session is still alive or dead. For example, when a wireless User Agent (UA) in conversation fails to connect to the network (e.g., due to abnormal radio disconnection), the SIP proxy server cannot recognize the failure of the session. Then the proxy server will hold the resources reserved for the failed session, which results in the blocking of other new session requests due to the lack of resources. To resolve this problem, one of the SIP Extensions, *SIP Session Timer*, specifies a keep-alive mechanism for SIP sessions [19]. In this mechanism, the duration of a communicating session is extended by using an `UPDATE` request sent from one SIP UA to the proxy server (then to the other SIP UA). A session timer (maintained in the proxy server and the UAs) records the duration of the session that the UA requests to extend. When the session timer nearly expires, The UA re-sends an UPDATE request to refresh the session interval. Note that the initial value for the session is carried in an `INVITE` request instead of `UPDATE`.

The existing approach to implement the SIP Session Timer mechanism are based on static (periodic) session refreshing. That is, the interval between two successive `UPDATE` requests (i.e., the length of the session timer) is set to a fixed value. The selection of the length for the session timer affects the system performance. A small session interval may cause excessive messaging traffic. On the contrary, if the session interval is large, excessive messaging traffic can be avoided while the proxy server will hold to resources for a longer period of time. From the above reasons, we observe that, the length of the session timer should be changed adaptively depending on the

network condition. Thus based on SIP Session Timer, we propose a dynamic session refreshing approach to adjust the session interval according to the network state to optimize the system performance.

A. SIP Session Timer

This section describes the SIP Session Timer mechanism defined in the SIP extension [19] , and then presents the dynamic refreshing approach based on the SIP Session Timer.

In SIP Session Timer, one of the UA in conversation sends an `UPDATE` request to extend the duration of the session. The interval between two consecutive `UPDATE`s (i.e., the length of the session timer) is determined through a negotiation between the UAC and the UAS. If an `UPDATE` request is not received before the session timer expires, the session is considered as abnormal disconnection, and the session will be force-terminated. Then the proxy server will release the allocated resources for the failed session.

To support SIP Session Timer, the following two header fields carried in the `UPDATE/INVITE` request (or in the corresponding response) are defined [19],

Session-Expires (SE) Header conveys the session interval for an established session. It specifies the interval between two successive `UPDATE` requests.

Min-SE Header indicates the minimum value for the session interval. This header is used to avoid excessive `UPDATE` requests sent to the proxy server (e.g., denial-of-service attacks) that affect the network performance. This header is carried in `422 Session Interval Too Small` response to indicate that the value in the *SE* header of the previous `UPDATE` (or `INVITE`) request is too small.

B. Dynamic Session Refreshing Approach

The dynamic session refreshing mechanism is developed for the VoIP architecture shown in Fig.6.

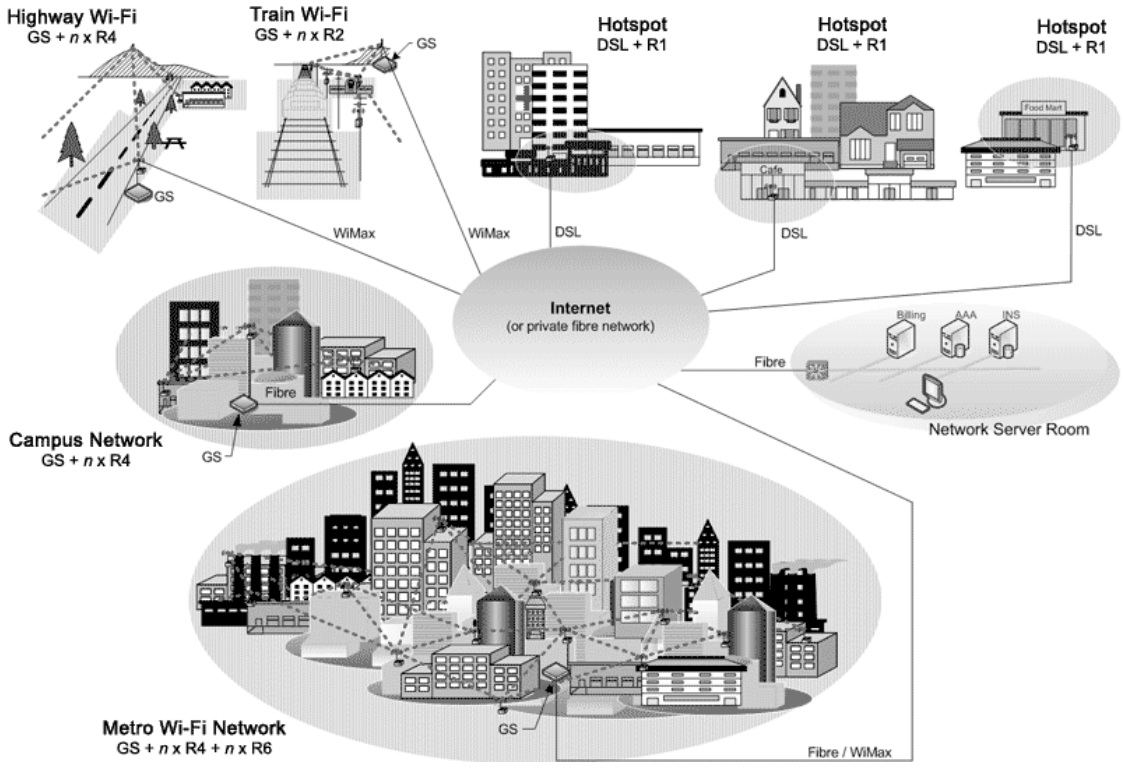

Fig. 6. SIP Call Architecture

In this figure, the UAs can access IP telephony services via heterogeneous networks including the wireless/mobile networks (e.g., IEEE 802.11 WLAN and GPRS/3G) and the wire line networks (e.g., cable, ADSL and PSTN). For an established session, abnormal detachment from the network for one of the participant UAs will result in the force-termination of the session. By using `SIP Session Timer` mechanism, the proxy server can detect the session force-termination and the proxy server can quickly release the resources allocated for the failed session.

The UA's abnormal detaching mainly results from the following reasons:

1. The failure of an UA

2. The radio disconnection for the wireless UA

The failure of an UA is not predictable and seldom occurs while the condition of mobile/wireless networks can be estimated based on the data collected from low layers (e.g., medium access control:MAC). We assume that the probability for the congestion/disconnection of wire line networks is negligible. To efficiently provide the *SIP Session Timer* mechanism, we propose a dynamic session refreshing approach to adaptively adjust the length of the session timer based on the condition of the wireless channel.

To estimate the state of the radio link for a wireless UA, the data from lower layers (e.g., MAC) should be periodically collected. This information is used to determine the issuance of the next `UPDATE` request. A low FER represents a "GOOD" network state with low probabilities of packet loss, and with low probabilities of the radio disconnection. If the FER is equal to or greater that a *Good Threshold (GT)*, then the network is said to be in "GOOD" state. In this case, to conserve the network bandwidth, the session timer is increased based on an *Increase Ratio (IR)* to avoid sending the UPDATE request frequently. If the network state has been good for a long time, the session interval will become very large, there by delaying the proxy server's reaction to session disconnection. Thus, to prevent the session timer from being too large, an *Upper Bound (UB)* for the session timer is set.

On the contrary, if the FER is large, (equal to or greater than the *Bad Threshold (BT)*) the network is said to be in "BAD" state. In this state, the established session

will fail with high probability. Thus, in order to detect the session failure earlier, the `UPDATE` requests should be sent more frequently by decreasing the session timer based on the *Decrease Ratio (DR)*. Similar to the UB, we need a *Lower Bound* LB to avoid congesting the network with UPDATE packets. Note that the value of LB is set to *Min-SE*.

When the network changes state rapidly between GOOD and BAD states, the session timer will oscillate between two values. In order to improve the performance of the dynamic session timer, we should have the change in the timer only for significant network change. The operation of the session timer is illustrated in the flow chart (Fig.7).

C. Analytical Model

Based on the VoIP architecture in Fig.8, we propose an analytical model to evaluate the performance of the SIP Session Timer for wireless VoIP. We assume the caller is mobile (such as IEEE 802.11 WLAN or 3G mobile data access) and the callee is connected through a wire line access. Also after the session is setup, the caller is responsible for issuing the UPDATE requests to the proxy to refresh the session interval. This model can be extended to having both the participants to be wireless users.

We model the condition of the wireless link for wireless user, with "GOOD", "BAD" and "DEAD" states [29] [30]. The different network states represent different FER for wireless links. In "GOOD" state, the FER is small, while in the "BAD" state, the FER is large. When the wireless network enters the "DEAD" state, the signaling path (between A and the proxy server) and the voice path (between A and B) are force-disconnected, and all the packet deliveries will fail.

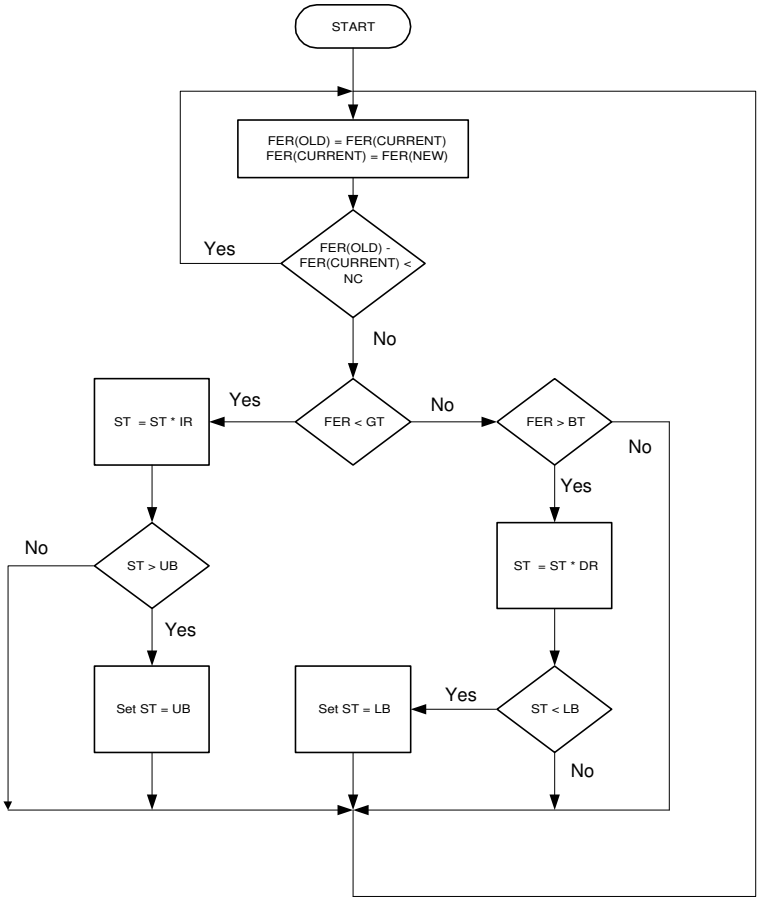

Fig. 7. Dynamic Session Timer Refresh Approach

The transition probabilities from "GOOD" state to "BAD" and "DEAD" states are respectively denoted by P_{gb} and P_{gd} $(P_{gb} + P_{gd} = 1)$. Similarly, the transition probabilities from "BAD" state to "GOOD" state and "DEAD" state are respectively denoted by P_{bg} and P_{bd} $(P_{bg} + P_{bd} = 1)$. The time intervals (i.e., t_g and t_b) that the wireless UA stays in "GOOD" and "BAD" states are assumed to have Exponential distribution with rates λ_g and λ_b respectively.

These transition probabilities are illustrated in Fig.9.

We assume that the packet loss probabilities for "GOOD" and "BAD" states are respectively P_{lg} and P_{lb}. The following input parameters are considered in our

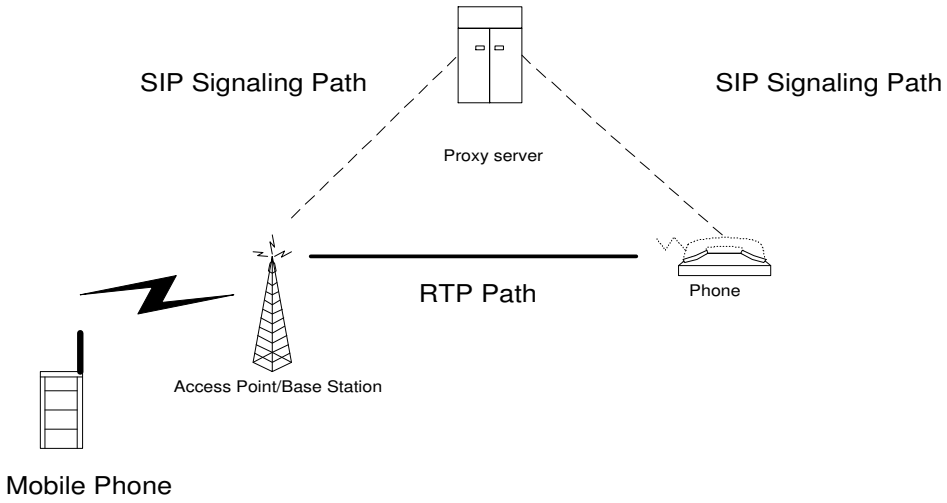

Fig. 8. Analytical Model

analytic model.

t_c: the call holding time with an exponential rate λ_c. We assume that the call termination is performed either by A or by B through the BYE request.

t_u: the time interval between two consecutive UPDATE packets for the wireless UA. We assume that t_u has an exponential distribution with rate λ_u.

P_s: the probability that the packet is successfully transmitted between the two endpoints. Thus, $P_s = \frac{(1-P_{lg})\lambda_b+(1-P_{lb})\lambda_g}{\lambda_b+\lambda_g}$.

Several output measures are defined in this study, and listed as follows.

p : the probability that the detection event (i.e., UPDATE loss) occurs before the call actually fails or completes.

n : the average number of UPDATEs issued for an established call.

r : the average response time i.e., the time interval between the time failure occurs and the time that the proxy server releases the resources for the call.

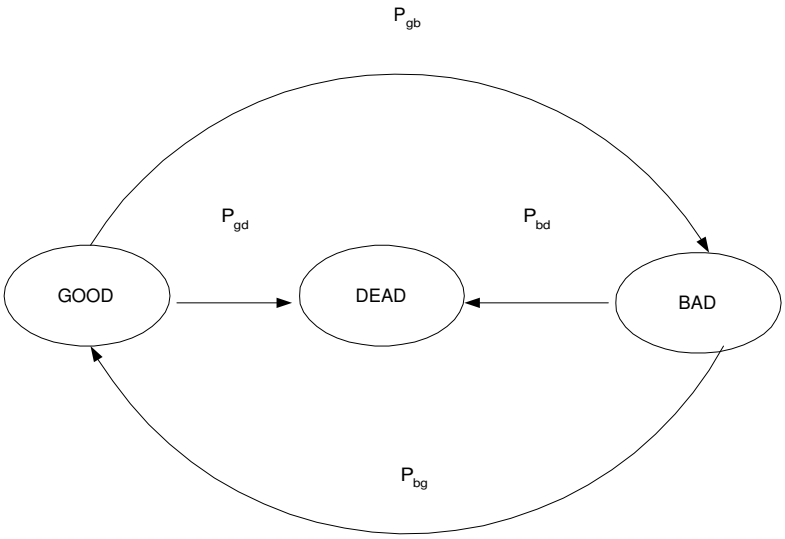

Fig. 9. Probability Transition Diagram between Network States

Based on the input and output parameters, the timing diagram is shown in Fig.10 below.

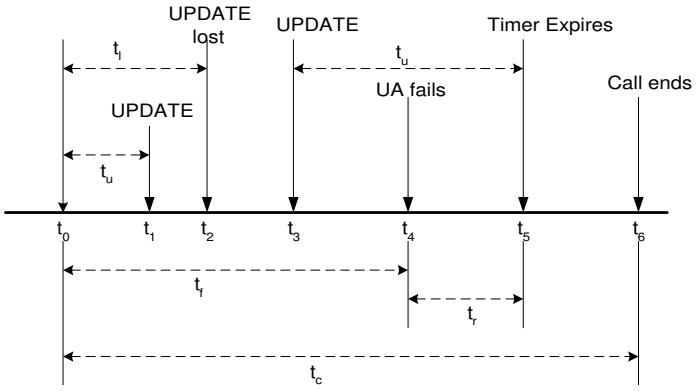

Fig. 10. Session Timer – Timing Diagram

We also take $t_d = \min(t_c, t_f)$, where t_f is the time to failure and t_c is the call duration time. In other words, t_d represents the interval between the time that the call is established and the time that the call is terminated due to failure or completion. Also, let $t_e = \min(t_c, t_f, t_l)$ be the interval between the time that call is established

and the time the call is terminated because of failure, completion, or mis-detection, where t_l is the mis-detection time. The distribution function F_d(T) for t_d will be derived in the following section. The derivation of F_e(T) is similar and the details are omitted.

D. Derivation of $F_d(T)$

This section elaborates of the derivation of $F_d(T)$. Based on the network state for the call arrival, two cases are considered and described below.

Case A: The call arrival is in "GOOD" state with probability $r = \frac{\lambda_b}{\lambda_b+\lambda_g}$. Then one of the following events occur

- The call enters the "BAD" state with probability $r_1 = \frac{\lambda_g P_{gb}}{\lambda_g+\lambda_c}$
- The call completes with probability $r_2 = \frac{\lambda_c}{\lambda_g+\lambda_c}$
- The call fails due to radio disconnection of A (i.e., "DEAD" state) with probability $r_3 = \frac{\lambda_g P_{gd}}{\lambda_g+\lambda_c}$

Case B: The call arrives in "BAD" state with probability $s = 1 - r$

- The call enters the "GOOD" state with probability $s_1 = \frac{\lambda_b P_{bg}}{\lambda_b+\lambda_c}$
- The call completes with probability $s_2 = \frac{\lambda_c}{\lambda_b+\lambda_c}$
- The call fails due to radio disconnection of A (i.e., "DEAD" state) with probability $s_3 = \frac{\lambda_b P_{bd}}{\lambda_b+\lambda_c}$

Let $F_{d,f}(T)$ be the cumulative distribution function of t_f given that $t_f < t_c$. To derive $F_{d,f}(T)$, the Table I, lists all possible situations for the case where the

failure occurs before the call completes, and their corresponding probabilities. Let $\lambda_1 = \lambda_g + \lambda_c$ and $\lambda_2 = \lambda_b + \lambda_c$, and t_1 and t_2 be exponentially distributed with rates λ_1 and λ_2, respectively. Let $t_{g,i} = \lfloor\ i/2 \rfloor\ t_1 + \lceil\ i/2 \rceil\ t_2$ and $t_{b,i} = \lceil\ i/2 \rceil\ t_1 + \lfloor\ i/2 \rfloor\ t_2$.

From the Table I, we have

$$F_{d,f}(T) = \frac{\Sigma_{i=1}^{\infty}[P_{d,f_i} F_{d,g_i}(T) + Q_{d,f_i} F_{d,b_i}(T)]}{P_f} \tag{4.1}$$

where, P_f is the probability that the call completes in failure and F_{d,g_i} and F_{d,b_i} are the distribution functions for $t_{g,i}$ and $t_{b,i}$ respectively. F_{d,g_i} is obtained through the convolution of the density functions of the two exponentially distributed random variables t_1 and t_2. The derivation for F_{d,b_i} is similar to F_{d,g_i}.

Let $F_{e,c}(T)$ be the distribution function of t_c when $t_c < t_f$. Hence, from Table II, we can derive $F_{e,c}(T)$ as

$$F_{d,c}(T) = \frac{\Sigma_{i=1}^{\infty}[P_{d,c_i} F_{d,g_i}(T) + Q_{d,c_i} F_{d,b_i}(T)]}{P_s} \tag{4.2}$$

where, P_s is the probability that the call completes in success.

Based on the above results, $F_d(T)$ can be derived as

$$F_d(T) = P_f F_{d,f}(T) + P_s F_{d,c}(T) \tag{4.3}$$

E. Output Measures

In this section, we derive the output measures p (the probability of detecting the non-arrival of the UPDATE message), n (the average number of UPDATE requests for a call, and r, (the average response time).

The probability of detection of non-arrival of packets should ideally take place before the end of the call to minimize the call resources wastage. The probability can

Table I. Possible situations and the corresponding probabilities for the case of $t_f < t_c$ with "GOOD","BAD" and "DEAD" abbreviated to "G","B" and "D"

Situation for "GOOD" Call Arrival	Probability
$G \to D$	$P_{d,f_1} = r.r_3$
$G \to B \to D$	$P_{d,f_2} = r.r_1.r_3$
$G \to B \to G \to D$	$P_{d,f_3} = r.r_1.s_1.r_3$
$G \to B \to G \to B \to D$	$P_{d,f_4} = r.r_1.s_1.r_1.r_3$
...	...
Situation for "BAD" Call Arrival	Probability
$B \to D$	$Q_{d,c_1} = s.s_3$
$B \to G \to D$	$Q_{d,c_2} = s.s_1.s_3$
$B \to G \to B \to D$	$Q_{d,c_3} = s.s_1.r_1.s_3$
$B \to G \to B \to G \to D$	$Q_{d,c_4} = s.s_1.r_1.s_1.s_3$
...	...

Table II. Possible situations and the corresponding probabilities for the case of $t_c < t_f$ with "GOOD","BAD" and "COMPLETION" abbreviated to "G","B" and "C"

Situation for "GOOD" Call Arrival	Probability
$G \rightarrow C$	$P_{e,c_1} = r.r_2$
$G \rightarrow B \rightarrow C$	$P_{e,c_2} = r.r_1.r_2$
$G \rightarrow B \rightarrow G \rightarrow C$	$P_{e,c_3} = r.r_1.s_1.r_2$
$G \rightarrow B \rightarrow G \rightarrow B \rightarrow C$	$P_{e,c_4} = r.r_1.s_1.r_1.r_2$
...	...
Situation for "BAD" Call Arrival	Probability
$B \rightarrow C$	$Q_{e,c_1} = s.s_2$
$B \rightarrow G \rightarrow C$	$Q_{e,c_2} = s.s_1.s_2$
$B \rightarrow G \rightarrow B \rightarrow C$	$Q_{e,c_3} = s.s_1.r_1.s_2$
$B \rightarrow G \rightarrow B \rightarrow G \rightarrow C$	$Q_{e,c_4} = s.s_1.r_1.s_1.s_2$
...	...

be computed only for the duration of the call (t_f or t_c whichever is smaller).

Given $t = min(t_f, t_c)$, then we can derive the probability p as

$$p = \int_{t=0}^{\infty} 1 - e^{-\lambda_u t(1-P_s)} dF_d(t) \tag{4.4}$$

or

$$p = \Sigma_{j=1}^{\infty}[F_d(\frac{j}{S}) - F_d(\frac{j-1}{S})][1 - e^{\lambda_u(\frac{j}{S})(1-P_s}] \tag{4.5}$$

Also, the UPDATE requests are issued only during the session. The call termination results from the following situations: the call completion, call failure and the mis-detection. If $t = min(t_l, t_c, t_f)$, then the average number of UPDATES issued n can be derived as

$$n = \int_{t=0}^{\infty} \lambda_u t dF_e(t) \tag{4.6a}$$

or

$$n = \Sigma_{j=1}^{\infty}[F_e(\frac{j}{S}) - F_d(\frac{j-1}{S})](\frac{j\lambda_u}{S}) \tag{4.6b}$$

The response time r is the time interval between the actual termination of the call by the wire line callee (or call failure)and the expiry of the session timer, which results in the termination of the call. Once failure occurs, the proxy server will release the resources for the call when one of the following events occurs.

- The session timer expires
- The wire line UA (i.e., the callee) terminates the call

Based on this information, the response time is calculated as

$$r = \frac{2\lambda_u + \lambda_c}{2\lambda_u(\lambda_u + \lambda_c)} \tag{4.7}$$

F. Performance Evaluation

Based on the analysis in the previous section, we use numerical examples to investigate the performance of n (average number of UPDATE requests for a call), p (i.e., the probability of detecting the non-arrival of UPDATE packet) and r (the response time) for the dynamic session refreshing approach. In this experiment, the default values for the parameters are set as follows: $\lambda_g = 3\lambda_c, \lambda_b = 5\lambda_c, P_{lg} = 10^{-6}, P_{lb} = 10^{-3}, P_{gd} = 10^{-6}, P_{bd} = 0.05, IR = 1.5, DR = 0.65, LB = \frac{1}{20\lambda_c}, UB = \frac{1}{5\lambda_c}$. Also the initial value for the session timer is set to $\frac{1}{10\lambda_c}$ and the query frequency for the radio link information is $30\lambda_c$.

Effect of λ_g:

The figures, plot the average number of UPDATE requests per call (n) (Fig: 11), the average response time (r) [Fig.12], and the mis-detection probability (p) [Fig.13] as a function of λ_g, where the input parameters except λ_g are set to the default values shown above.

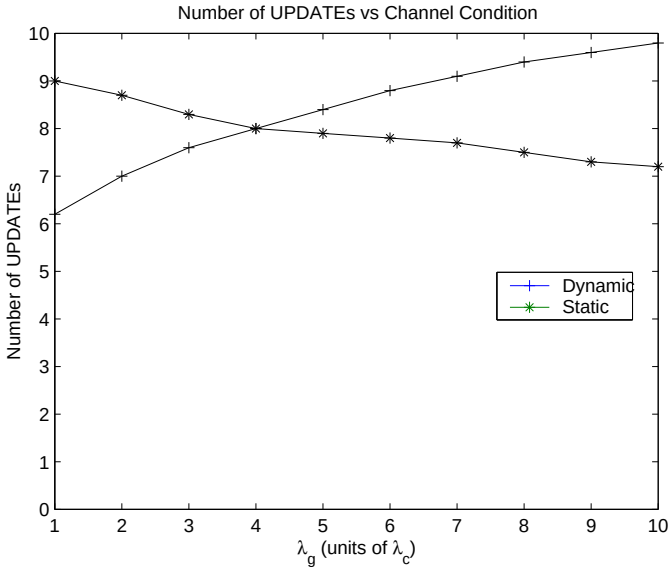

Fig. 11. Number of UPDATES, n

In Fig. 11, as λ_g increases(i.e., the reduction of the average time of the good state where a wireless UA resides), the curves for the static and dynamic session refreshing approaches respectively, decrease and increase. For $\lambda_g < 4\lambda_c$, the static refreshing approach has more UPDATE requests than the dynamic one. On the other hand, where λ_g is larger than $4\lambda_c$, the opposite result is observed.This can be explained as follows. As λ_g increases, the average time in the bad state for a call relatively increases. Thus, the call suffers from the radio disconnection more probably, and the call holding time (i.e., the average duration of the call, t_c) decreases due to the increasing likelihood of force-termination. For the static session refreshing approach, the UPDATE request is periodically sent regardless of the network state. However, the call holding time (t_c)decreases, n for the static approach decreases (since fewer `UPDATE`s are sent for a shorter call). On the contrary, the frequency of UPDATE deliveries for our dynamic approach increases when the network state remains bad, and this results in the increase of the session refreshing number n.

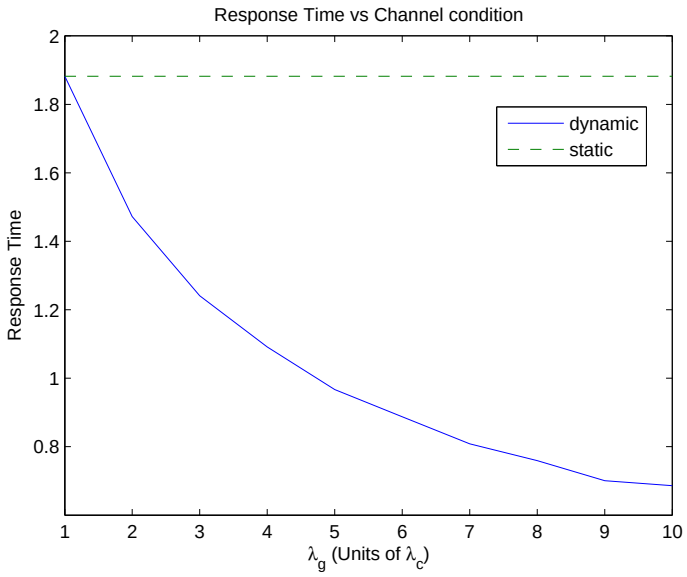

Fig. 12. Response Time, r

Fig. 12 shows that the response time for the static refreshing approach is not influenced by λ_g However, for the dynamic refreshing approach, r significantly decreases as λ_g increases, which indicates that the dynamic approach effectively adjusts the session timer especially when the network condition is BAD. Furthermore, as shown in Fig.13,

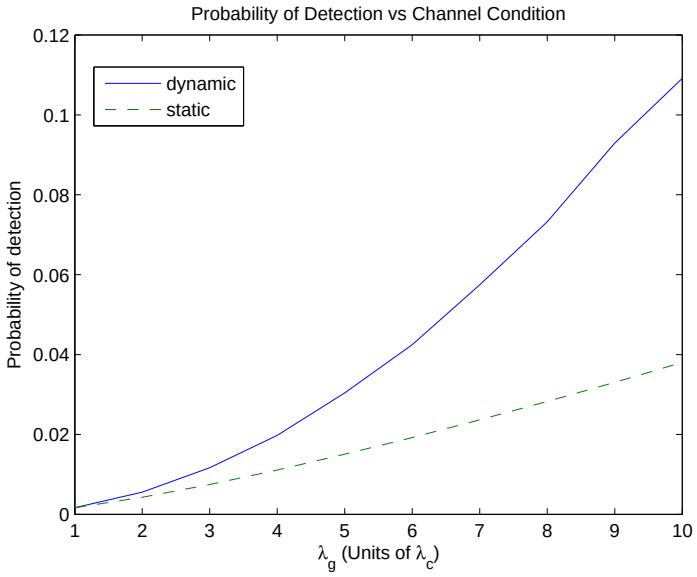

Fig. 13. Probability of Detection, p

The probability of mis-detection, p is an increasing function of λ_g for both static and dynamic session refreshing approaches. The mis-detection results from the UPDATE loss prior to the actual call failure or completion. Asλ_g increases, the UPDATE request gets more probably lost due to the increase in the duration of the bad state of the call, and thus the mis-detection probability increases. However, due to the variation of the frequency of the `UPDATE`s in the dynamic case, the probability of mis-detection is higher than the static case.

The next chapter will provide an overview on the handover mechanisms using SIP and Mobile IP , and discuss the implementation of an hybrid handover.

CHAPTER V

HYBRID HANDOFF

In this chapter we discuss a hybrid handoff scheme, which is the hybrid of Mobile IP and SIP. This hybrid scheme reduces the handoff delay for real time applications. It avoids the need for tunneling the packets throughout the handoff period and the communication need not be wait till a new IP is given by the DHCP server.

We shall provide an overview on SIP and Mobile IP handoff schemes before detailing the hybrid algorithm.

A. SIP Handoff

SIP Handoff is an application layer handoff. The advantage of an application layer handoff is that, there is complete transparency regarding the implementation of the lower layers. Let us see how does an application layer handoff take place.

We can assume that the mobile host belongs to some home network, on which there is a SIP server, which receives registrations from the mobile host each time it changes location. The mobile host does not need to have a statically allocated IP address on the home network. When the correspondent (static) node sends an `INVITE` to the mobile host, the SIP server has current information about the mobile host's location and redirects the `INVITE` there.

If the mobile host moves during a session [14], it must send a new `INVITE` to the correspondent node using the same call identifier as in the original call setup. It should put the new IP address (obtained from the DHCP server at the new location) in the `Contact` field of the SIP message, which tells the correspondent host where it wants to receive future SIP messages. To redirect the data traffic flow, it indicates

the new address in the SDP field, where it specifies transport address.

Thus, the SIP based handover is dependent on the speed with which the DHCP server serves the IP address, and the time taken for processing the re-`INVITE` message. If we assume the time taken for processing to be negligible, then the DHCP server response time is critical for handover.

B. Mobile IP Handoff

Mobile IP is a transparent solution that handles mobility at the network layer. This scheme was designed to solve the mobility problem by allowing the mobile node to use two IP addresses: a fixed home address and a care of address that changes at each new point of attachment. This solution is a transparent because, when two nodes communicate with each other and one of them moves to a new subnet, then the other node is completely unaware of this mobility, and it continues its communication as if nothing has happened. In Mobile IP, it is assumed that every mobile node (MN) has its home network, and a statically allocated IP address on its home network. In Mobile IP, the support for mobility is provided by adding IP tunneling to IP routing.

The Mobile IP architecture mainly consists of

1. Mobile Node (MN)
2. Correspondent Node (CN): The CN participates in communication with the MN. The CN can be either fixed or a mobile node.
3. Home Agent (HA): The default router on the home network, HA stores the current locations of all the mobile nodes in its network. It intercepts the packets from CN and tunnels them to the current location of the mobile node.
4. Foreign Agent (FA): The default router on the foreign network, FA receives the

tunneled packet from HA and forwards them to the MN in its network.

1. Main Functions in Mobile IP

Mobile IP consists of three main functions [31],

1. Mobile Agent Discovery: When a mobile node is away from its home base, it needs to find agents so as to maintain internet access.

2. Registration with home agent: The mobile node registers its care-of address with its home agent in order to obtain service.

3. Packet delivery using IP tunneling: The mobile node gets forwarded packets from its foreign agents which were originally sent from the mobile node's home agent. Tunneling is the method used to forward the message from home agent to foreign agent and finally to the mobile node.

2. How Mobile IP Works

The Home agents (HA) and the foreign agents (FA) will broadcast agent advertisements in their subnets. The mobile node detects that it has moved to a new subnet, when it hears the advertisement from an agent that is not its HA. Then it sends a registration request to its HA through the FA with a care-of address (CoA), which is generally the address of the foreign agent itself. The HA then replies either accepting or denying that request. The CN will be completely unaware of this mobility and will transmit the packets to the mobile node's home address. The HA will intercept these packets and performs the IP-in-IP encapsulation and tunnels them to the CoA of the mobile node (FA). The FA then decapsulates the packets and forwards them to the mobile node.

C. Hybrid Handoff

We shall now detail the Hybrid handoff scheme.

If the Correspondent Node (CN) wants to communicate with the Mobile node (MN), they start the communication using the SIP call establishment mechanism. If the MN moves to a new subnet, with the session being still active, then, the MN uses the Mobile IP scheme and receives the packets in the new subnet with tunneling being done by its *Home Mobility Agent*(HMA). The tunneling is required only until the MN receives a new IP address from the DHCP server in the new subnet. As soon as the MN gets the new IP, it sends the SIP RE-`INVITE` message, with the same Call-ID, to CN informing it about its new IP. From then on, tunneling is stopped, and the packets will be sent to the current location of the MN directly.

The detailed working of this scheme is illustrated in Fig.14.

If Correspondent Node (CN) wants to communicate with Mobile Node (MN), then

1. CN sends an `INVITE` message to the Mobility Agent (MA) in MNs home domain.

2. The Mobility Agent performs a DNS lookup for the current location of the MN, and gives this information to the CN.

3. CN sends the `INVITE` message directly to the current location of MN, and the call is established.

   Meanwhile, if the MN moves to a new subnet during the ongoing communication with CN, resulting in a handoff, then,

4. MN sends a Registration request to its Home Mobility Agent (HMA) through Foreign Mobility Agent (FMA).

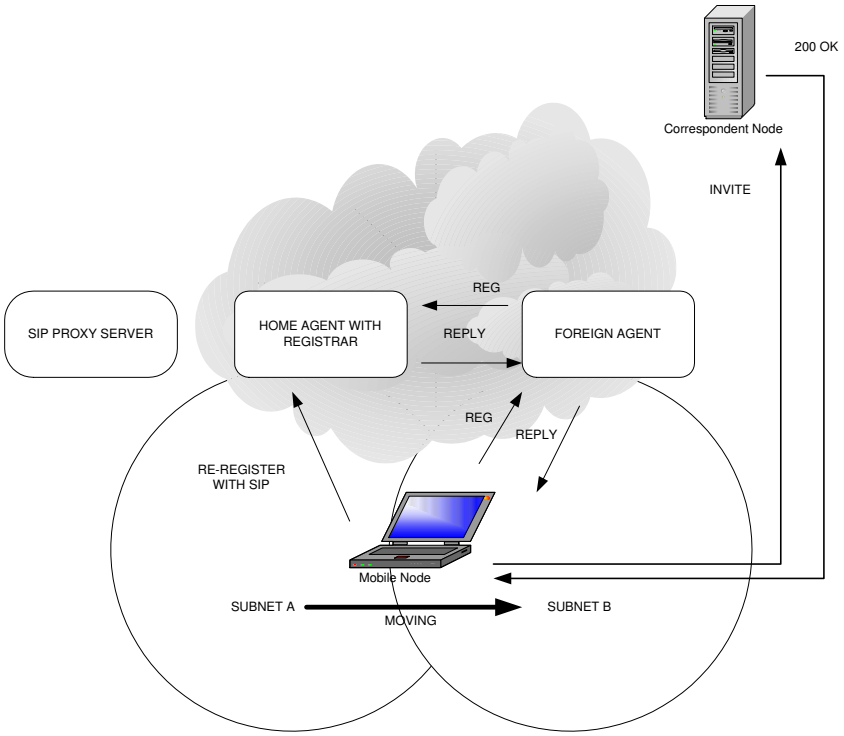

Fig. 14. Hybrid Handoff

5. MN gets the reply from its HMA accepting its registration.

6. Soon after this, the transmission resumes, as in normal Mobile IP scheme.

7. The packets will be tunneled to the MN through FMA by the HMA.

8. Meanwhile, the MN contacts DHCP server to get a new IP address.

9. After it gets a new IP, it sends the SIP RE-`INVITE` message to the Correspondent Node (CN), with its new location.

10. From then on, the packets will be sent directly to the MNs new current location without tunneling.

11. The MN sends a SIP RE-`REGISTER` message to its HMA, so that any new connections in future can be correctly redirected to the MNs current location.

The messages involved in the handoff process are given in Fig.15.

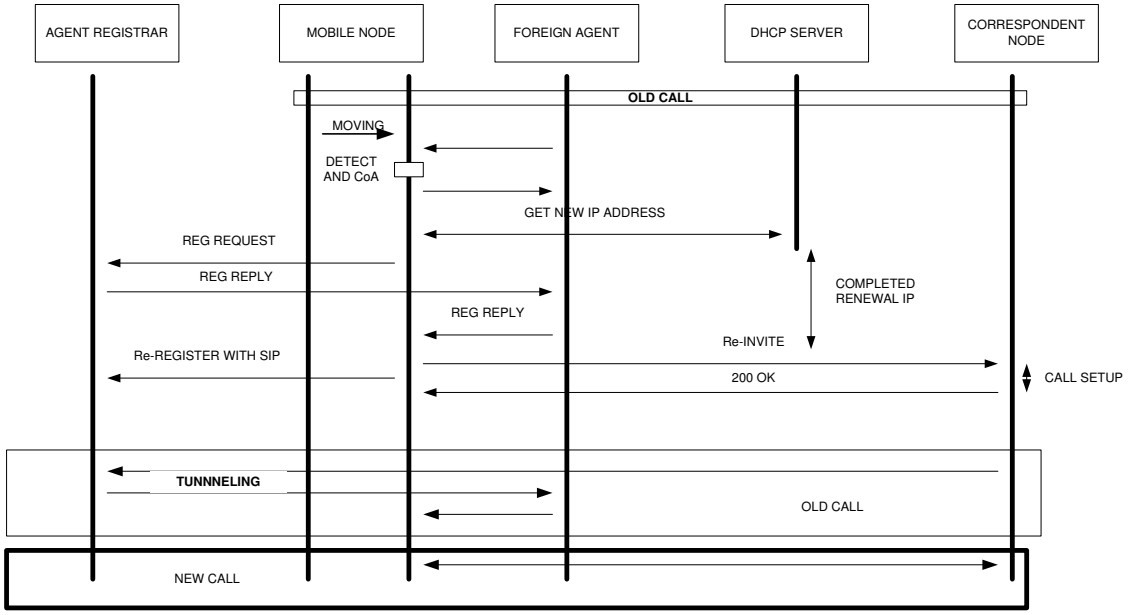

Fig. 15. Hybrid Handoff Messages

This scheme overcomes the problems associated with Mobile IP and SIP. The communication resumes as soon as the node hears the agent advertisement in the foreign network as it happens in Mobile IP, without waiting until the node obtains the new IP address from the DHCP server as in SIP scheme. Also, the packets need not be tunneled through the longer path, for the entire session as in Mobile IP.

D. Implementation

The SIP handles the mobility at the application layer. The SIP patch has been downloaded from the National Institute of Standards Technology web site [32]. The patch is applied on the NS2.1b9a version [33]. The downloaded SIP patch was for the wired scenario and mobility was not supported. In order to support mobility, DHCP agents had to be included in NS. DHCP Agent would be responsible for allotting new

IP addresses, when requested for, to the mobile nodes in the foreign domains.

1. Simulation Setup

The following assumptions were made during the simulation.

1. The DHCP delay, time taken for obtaining a new IP address from the DHCP server, is taken as 2 seconds. This value is taken after measuring the DHCP delay at Internet2 Technology Evaluation Center, Texas A&M University, College Station.
2. The DHCP server is present on the base station node itself.
3. The authentication delays are not considered in our simulations.
4. Mobile IPv4 is used for the simulation.

2. Scenario

In the simulation, all the nodes are placed in an area of size 670m x 670 m. There are three domains, the home domain, the foreign domain and the wired domain. There are three nodes, one in each domain. The topology is shown in the Fig.16.

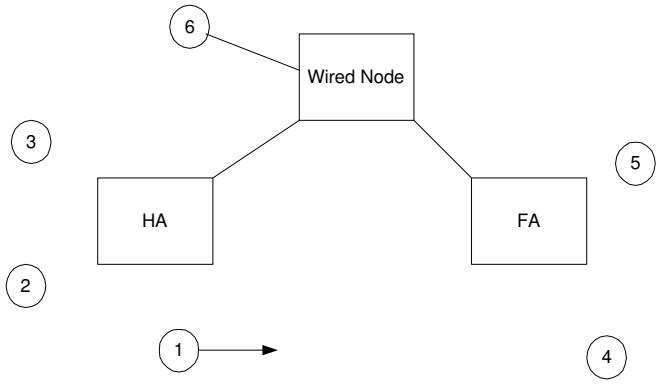

Fig. 16. Wireless Topology

In this scenario, the Home Mobility agent is on the node HA which acts as both the Mobile Nodes home agent and the SIP redirect server. Here RTP traffic is set up on the SIP connection, with a packet size of 500 bytes. The mobile node **1** moves to the foreign domain in the middle of their communication, where FA is the mobility agent.

From Fig.17, one can observe that the period during which the packets are sent using Mobile IP. In the figure, this period is around 50 packets. The total tolerable delay for real time communications is around 500 ms. From the figure we understand that we remain under 200 ms during the period of handoff. Actually this delay during Mobile IP depends upon the distance between the HA and MN.

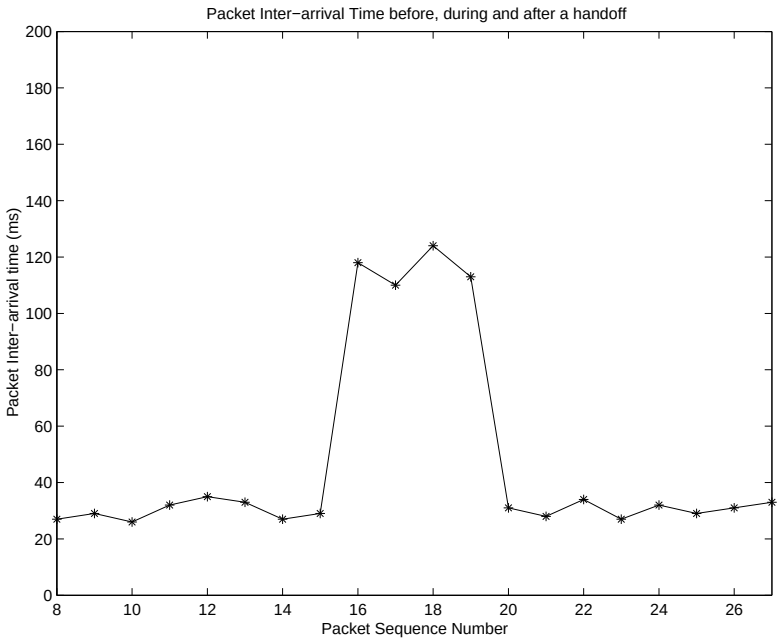

Fig. 17. Packet Inter-arrival Time

The performance of the Mobile IP and the hybrid scheme depends heavily on the distance between the Home Agent (HN) and the mobile node. All the simulations are performed using the network topology shown in Fig.16. In these simulations, we

are interested in the packet loss during call in progress and the handoff latency. The CN acts as a RTP traffic source, producing fixed length packets (500 bytes). The MN acts as a sink. The results are observed in Fig.18.

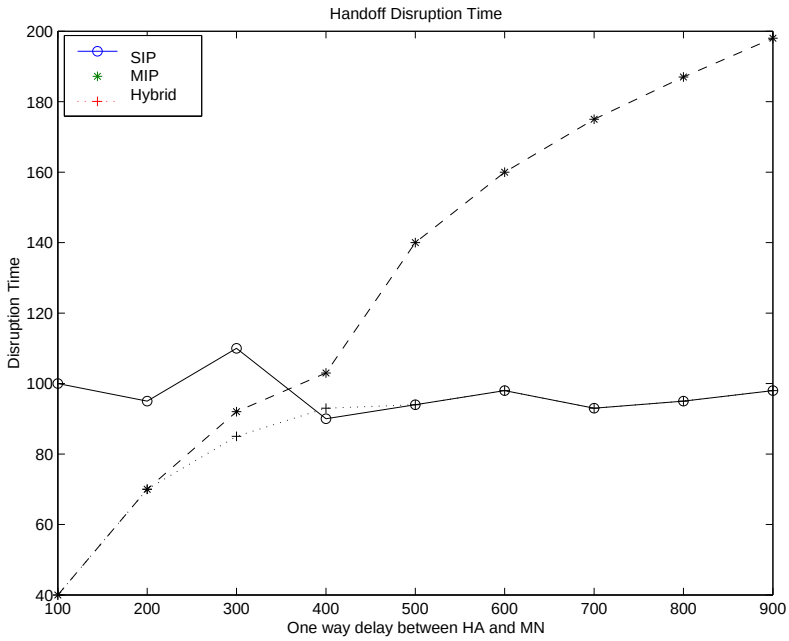

Fig. 18. Call Disruption Time during Handoff

From Fig.18 , it is clear that the disruption during the handoff with SIP-mobility becomes shorter than that of Mobile IP approach as the distance between the MN and its home network increases, since the SIP mobility handoff depends mainly on the distance between CN and MN. However, the disruption time in MIP increases directly proportional to the distance between the HA and MN, since the MN must now register its IP address with HA whenever it moves to a new sub-net. Finally, the integrated mobility management approach shows the most effective results since it benefits both from Mobile IP, when the MN to home network delays are small, and SIP when the updating the CN is faster than updating the home network agents.

CHAPTER VI

CONCLUSIONS

In this thesis , certain modifications were made to the extensions of Session Initiation Protocol for wireless channels. The Session Startup Time and the SIP session timer was modified to suit the channel conditions of wireless channels. In addition to this, a hybrid hand over technique is proposed for VoIP calls. The following paragraphs summarize the results of this research.

We have evaluated the average SIP session startup time depending on the FER of the wireless link. For FER greater than 1 %, the session startup time increases dramatically for both TCP and UDP. The use of a lower layer re-transmission mechanism such as RLP improves the session setup delay. In high FER environments, the session setup delay with RLP is comparatively small . Also, the use of error correcting mechanism or hybrid ARQ could improve the session startup time further.

The *SIP Session Timer* mechanism was proposed to track the states of the communicating sessions for proxy servers. Upon the occurrence of the session failure (e.g., radio link disconnection), the proxy server can quickly release resources allocated for the failed radio session by using *SIP Session Timer*. Based on *SIP Session Timer*, we propose a dynamic session refreshing approach to adjust the session timer depending on the conditions of the radio link. With this dynamic session refreshing approach, session failure can be detected without considerable increase in traffic. An analytical model was developed to investigate the performance of the static and dynamic session refreshing approaches. Our study indicates the following,

- The response time for the dynamic session refreshing approach significantly decreases as λ_g increases, while the response time for the static one is not influenced by λ_g. Furthermore, for all λ_g values under investigation, the dynamic

approach has smaller response times than the static value.

- As λ_g increases, the call holding time decreases due to the increasing force-termination probability. This results in the decrease in the number of `UPDATE`s for the static session refreshing approach. However, for dynamic session refreshing, the frequency of `UPDATE` deliveries increases when the network state remains bad, which results in the increase in the number of `UPDATE`s .

The hybrid scheme performs better than Mobile IP and SIP. By better performance,we mean lesser handoff delay. The duration of the handoff period would be significantly reduced if the mobile host could immediately begin using the new IP address while verifying in the background that it is not in use by any other host. Further work needs to be done to resolve some issues regarding authentication.

REFERENCES

[1] J. Harris and M. Airy,"Analytical model for radio link protocol for IS-95 CDMA systems," in *IEEE Vehicular Technology Conference Proceedings*, 2000, vol. 3, pp. 229–237.

[2] G .A. Thom, "H.323: The multimedia communications standard for local area networks," *IEEE Communications Magazine*, vol. 34, pp. 52–56, Dec 1996.

[3] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson et al. "SIP: The Session Initiation Protocol," IETF RFC 3261, June 2002.

[4] R. Fielding, G. Gettys, J. Mogul, H. Frystyk, L. Masinter et al., "Hypertext Transfer Protocol – HTTP/1.1," IETF RFC 2616, June 1999.

[5] J. Klensin, Ed., "Simple Mail Transfer Protocol," IETF RFC 2821, April 2001.

[6] S. Das, A. Mcauley, A. Dutta, A. Misra, K. Chakraborty, "IDMP: An Intra-Domain Mobility Management Protocol for next-generation wireless networks," *IEEE Wireless Communications*, vol. 9, no. 3, pp. 38-45, June 2002.

[7] N. Banerjee, W. Wu, K. Basu, S. K. Das, "Analysis of SIP-based mobility management in 4G wireless networks," *Computer Communications*, pp. 697–707, 2004.

[8] Y. Raivio, "4G Hype or Reality," in Proceedings of the IEE International Conference on 3G Mobile Communication Technologies, London,UK, March 2001, pp. 346–350.

[9] D. Raychaudhuri, *"4G network architectures: WLAN hot-spots, infostations and beyond,"* in International 4G Forum, London, UK, May 2002.

[10] T. La. Porta, R. Ramjee, L. Lee, L. Salgerelli,S. Thuel,"IP-based access network infrastructure for next-generation wireless data networks," *IEEE Personal Communications*, 2000, vol. 7, pp. 34–41.

[11] U. Varshney, R. Jain, "Issues in emerging 4G wireless networks," *IEEE Computer Magazine*, pp. 94-96, June 2001.

[12] J. -O. Vatn, "The effect of using co-located care-of addresses on macro handover latency, " in *Proceedings of 14th Nordic Tele-traffic Seminar*, (Technical University of Denmark, Lyngby, Denmark), Aug. 1998.

[13] A.G. Valk, "Cellular IP: A new approach to Internet host mobility," *ACM Computer Communications Review*, vol. 29, pp.50-65, Jan 1999.

[14] H. Schulzrinne, E. Wedlund, "Application-layer mobility using SIP," *ACM SIGMOBILE Computing and Communications Review*, vol.4, no. 3, pp. 47–57, July 2000.

[15] G.A. Mills-Teffey and D. Lotz, "Mobile voice over IP (MVoIP): An application-level protocol for call," in *IEEE International Conference on Performance, Computing, and Communications*, Phoenix, Arizona, pp. 271-279, April 2002.

[16] A. Dutta, F. Vakil, J.-C. Chen, M. Tauil, S. Baba, N. Nakajima, H. Schulzrinne, "Application layer mobility management scheme for wireless internet," *3G Wireless International Conference*, San Francisco, May 2001.

[17] N. Banerjee, W. Wu, S.K. Das, S. Dawkins, and J. Pathak, "Mobility support in wireless internet," *IEEE Wireless Communications*, vol. 10, pp. 54–61, Oct 2003.

[18] J. Rosenberg, "The Session Initiation Protocol (SIP) UPDATE Method," IETF RFC 3311, September 2002.

[19] S. Donovan, J. Rosenberg, "Session Timers in the Session Initiation Protocol (SIP)," IETF RFC 4028, April 2005.

[20] J. Lennox, "Services for Internet Telephony," Ph.D. Dissertation, 2004, Columbia University, http://www1.cs.columbia.edu/~lennox/thesis.pdf; accessed July 31, 2006.

[21] N. Freed and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME)," IETF RFC 2045, November 1996.

[22] M. Handley and V. Jacobson, "SDP: Session Description Protocol," IETF RFC 2327, April 1998.

[23] P. Hoffman, L. Masinter, and J. Zawinski, "The `mailto` URL scheme,", IETF RFC 2368, July 1998.

[24] H. Schulzrinne, "The `tel` URI for telephone numbers," IETF RFC 3966, December 2004.

[25] "Radio Link Protocol (RLP)," http://en.wikipedia.org/wiki/Radio_Link_Protocol; accessed July 31, 2006.

[26] W. Simpson, Ed., "The Point-to-point Protocol (PPP)," IETF RFC 1661, July 1994.

[27] G. Bao and A. Chockalingam,"Performance of TCP/RLP protocol stack on correlated fading DS-CDMA wireless links," *IEEE Transactions on Vehicular Technology*, vol 49, no. 1, pp. 28-33, Jan 2000.

[28] "Ethereal," http://ethereal.com; accessed July 31, 2006.

[29] Y.-B. Lin, Y.-R. Huang, A.-C. Pang, and I. Chlamtac,"All-IP approach for third generation mobile networks," *IEEE Network*, vol. 5, no. 16, pp. 8-19, 2002.

[30] Y.-B. Lin, Y.-R. Huang, A.-C. Pang, P. Agarwal, and I. Chlamtac, "Serving radio network controller relocation for UMTS All-IP Network," *IEEE Journal on Selected Area in Communications*, vol. 22, no. 4, pp. 617–629, 2004.

[31] C.E. Perkins and K.-Y. Wang, "Optimized Smooth Handoffs in Mobile IP," *IEEE Symposium on Computers and Communication Proceedings*, Red Sea, Egypt, p. 340, July 1999.

[32] "NS-2 SIP simulation patch", http://www-x.antd.nist.gov/proj/iptel/ns2-sip-simulation/ns-2.1b9a.sip.patch;accessed on July 31, 2006.

[33] "ns2: Network simulator," http://www.isi.edu/nsnam/ns/index.html; accessed on July 31, 2006.

VITA

Vijay Sundar Rajaram received his Bachelor of Engineering degree in electrical and electronics engineering from Bharathiyar University, PSG College of Technology in 2003. He entered the electrical and computer engineering program at Texas A&M University in August 2004, and since then he has been pursuing his Master of Science degree receiving it in December 2006. His research interests include wireless communication and next generation telephony systems.

Mr. Rajaram may be reached at 214, Zachry Engineering Center, Texas A&M University, College Station, TX-77801. His email address is vijaysundar@gmail.com

The typist for this thesis was Vijay Sundar Rajaram.